NOTE: Use these buttons to print or save the form. DO NOT use the browser tool bar. SAVE PRINT

LA-UR-07-3181

Approved for public release;
distribution is unlimited.

Title:	QUIC URB v. 1.1 Theory and Users Guide
Author(s):	Eric R. Pardyjak Michael J. Brown
Intended for:	Open Distribution

Form 836 (7/06)

SAVE PRINT CLEAR FORM

QUIC-URB v1.1
Theory and User's Guide

June 2, 2003
Revised 4-27-07

Eric R. Pardyjak and Michael Brown
Los Alamos National Laboratory

Note: this document was written for an early version of QUIC-URB (v. 1.1) and has not been updated for the new features found in the current version of QUIC. However, many of the basics of the model are similar. We intend to put out a revised version of this theory and user guide in the near future.

New features:

* The code is now written in FORTRAN 90 and works in single precision (faster and less memory)

* QUIC-URB is no longer limited to grid sizes of 1 meter. The user can enter a real world grid size (dx) and all output data is appropriately scaled. dx, dy, and dz can all be different sizes as well.

* In addition to the power-law and log-law inflow profiles, QUIC-URB now allows the user to specify an urban canopy inflow profile or a user-specified profile. In addition, the outer grid wind field can now be used to drive the inner grid domain.

* A subdomain may be defined that specifies where empirical wakes and cavities may be applied. Outside of this subdomain only the boundary layer parameterization is applied. As a result, the solution is essentially potential flow in these regions. This is intended to speed up the simulation for larger problems.

* QUIC-URB now allows building polygons to be stacked on top of one another.

* QUIC-URB can now handle cylindrical and ellipsoidal building shapes, and also has a pentagon-shaped building. Vegetation cells have also been added.

* New rooftop schemes have been added, so that there is now a choice of slip flow (original scheme), log-law, and recirculation (rooftop reverse flow).

See Pol S., N. Bagal, B. Singh, M. Brown, and E. Pardyjak, 2006: Implementation of a rooftop recirculation parameterization into the QUIC fast response wind model, 6th AMS Urb. Env. Symp., Atlanta, GA, LA-UR-05-8631, 19 pp.

* A new upwind scheme called modified vortex parameterization (MVP) has been implemented. Either the original Rockle scheme and the MVP scheme can be selected.

See Bagal, N., E. Pardyjak, and M. Brown, 2003: Improved upwind cavity parameterization for a fast response urban wind model, AMS Conf. on Urban Zone, Seattle, WA, 3 pp.

* The Rockle street canyon vortex scheme has been slightly modified to use the Fackrell cavity length formula in the street canyon determination.

* A new intersection scheme using a Poisson solver has been incorporated.

Introduction

As part of the DOE Chemical Biological Non-Proliferation (CBNP) program and more recently the DHS Biological Countermeasures program, Los Alamos National Laboratory has been working on fast response modeling in urban areas. These models enable information on the transport and dispersion of agents to be obtained quickly. To this end, a three-dimensional mass consistent diagnostic wind model for fast response in an urban environment is being developed.

This document describes QUIC-URB, part of the QUIC (Quick Urban & Industrial Complex) Dispersion Modeling System, a diagnostic wind model (written in FORTRAN77) that calculates 3D wind fields for flow around arrays of buildings. QUIC-URB v1.1 allows a user to input multiple rectangular shaped buildings at regular spacing on a three dimensional grid. The code can be run on any machine that has a FORTRAN77 compiler, however some of the make file options may need to be modified when moving from one compiler to another. Makefiles are currently available that compile using g77 or the free "Gnu" FORTRAN compiler on LINUX or UNIX machines as well as the Portland Graphics compiler on PCs.

Note that there is a graphical user interface (QUIC-GUI) that can be run through Matlab. This GUI can be used directly with input and output files from QUIC-URB and allows for easy and fast visualization of velocity vectors, streamlines, velocity profiles and contours (among other things).

1.0 Model Theory

The QUIC-URB model is based on the fast response urban wind modeling done in Rockle's (1990) thesis. In this model, an initial uniform wind field is prescribed (u_o,v_o,w_o) based on an incident flow, u_{in} and the various flow effects associated with building geometries. The final velocity field (u,v,w) is obtained by forcing the initial velocity field to be mass consistent. The resulting complex 3D velocity field resembles a time averaged experimental result. The initialization of the velocity field is shown in Figures 1 and 2 below. In regions where there are buildings, the velocities are forced to be zero (See Figure 1). Around the buildings themselves empirical parameterizations are invoked to produce a velocity field that maintains important features of the time averaged flow (for example see the wake cavity in Figure 2).

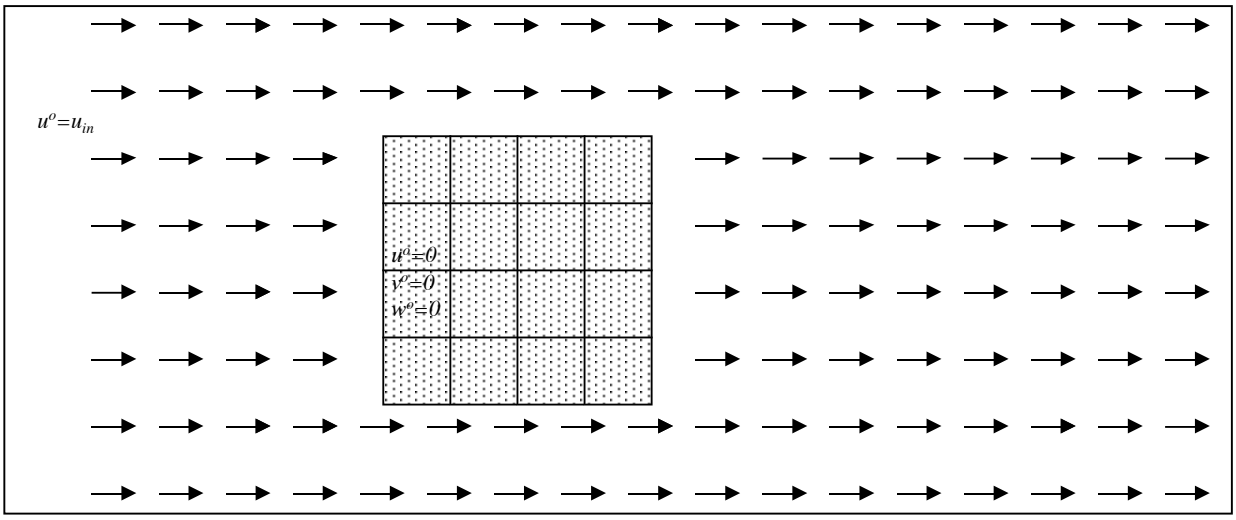


Figure 1 - Plan view (*x*-*y* plane) of a sample initial wind field around a cube.

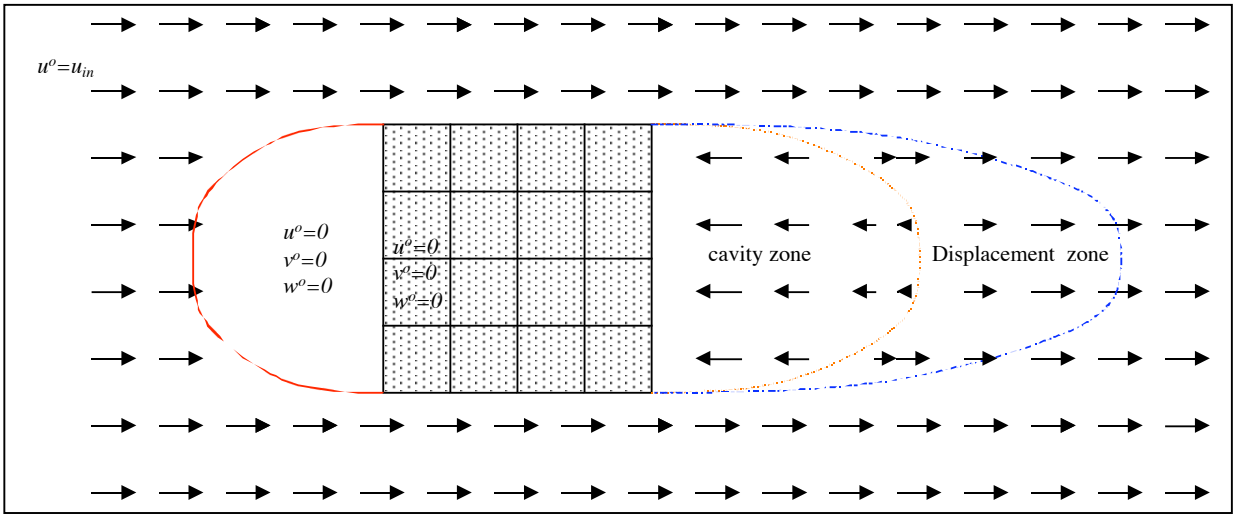


Figure 2 - Plan view (*x*-*y* plane) of initial wind field with empirical parameterizations.

The method used for QUIC-URB (version 1.0) employs the same empirical logic as Rockle (1990) (and later Kaplan and Dinar, 1996), however the numerical implementation is slightly different and will be discussed below. The objective of these notes are to clearly show how the Rockle method is fully implemented.

1.1 Numerical Method

This section outline the details of the numerical method used in QUIC-URB and how the methods are implemented.

Minimize the variance of the difference between the observed initial wind field (u_o,v_o,w_o) and analyzed wind field (u,v,w) subject to the physical constraints which are satisfied exactly or approximately by the analyzed wind field (Sherman, 1978).

$$E(u,v,w,\lambda) = \int_V \left[\alpha_1^2 \left(u-u^o\right)^2 + \alpha_1^2 \left(v-v^o\right)^2 + \alpha_2^2 \left(w-w^o\right)^2 + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial x} + \frac{\partial w}{\partial x}\right)\right] dxdydz \tag{1}$$

The main part of the code solves the following equation for the Lagrange multipliers λ, using an SOR method.

$$\frac{\partial^2 \lambda}{\partial x^2} + \frac{\partial^2 \lambda}{\partial y^2} + \left(\frac{\alpha_1}{\alpha_2}\right)^2 \frac{\partial^2 \lambda}{\partial z^2} = R \tag{2}$$

As shown in Figure 3, λ is a cell-centered quantity and the velocities are cell-faced quantities. The α's are Gaussian precision moduli and will be discussed Later.

For the cells unaffected by boundary's, Equation (1) is discritized as follows:

$$\frac{\lambda_{i+1,j,k} - 2\lambda_{i,j,k} + \lambda_{i-1,j,k}}{\Delta x^2} + \frac{\lambda_{i,j+1,k} - 2\lambda_{i,j,k} + \lambda_{i,j-1,k}}{\Delta y^2} + \left(\frac{\alpha_1}{\alpha_2}\right)^2 \frac{\lambda_{i,j,k+1} - 2\lambda_{i,j,k} + \lambda_{i,j,k-1}}{\Delta z^2} = R_{i,j,k} \tag{3}$$

for cells with solid boundaries in the positive i direction, the following simplification is made (define this boundary as a wall to the right):

$$\frac{\partial^2 \lambda}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial \lambda}{\partial x}\right) \approx \frac{1}{\Delta x}\left(\frac{\partial \lambda}{\partial x}\bigg|_{i+\frac{1}{2}} - \frac{\partial \lambda}{\partial x}\bigg|_{i-\frac{1}{2}}\right) \tag{4}$$

Since the boundary condition for solids is $\frac{\partial \lambda}{\partial x} = 0$, if the wall is at i+1/2, (3) simplifies to

$$\frac{\partial^2 \lambda}{\partial x^2} \approx \frac{1}{\Delta x}\left(0 - \frac{\partial \lambda}{\partial x}\bigg|_{i-\frac{1}{2}}\right) \approx \frac{1}{\Delta x}\left(-\frac{\lambda_i - \lambda_{i-1}}{\Delta x}\right) \quad \text{or,}$$

$$\frac{\partial^2 \lambda}{\partial x^2} \approx \frac{\lambda_{i-1} - \lambda_i}{\Delta x^2} \tag{5}$$

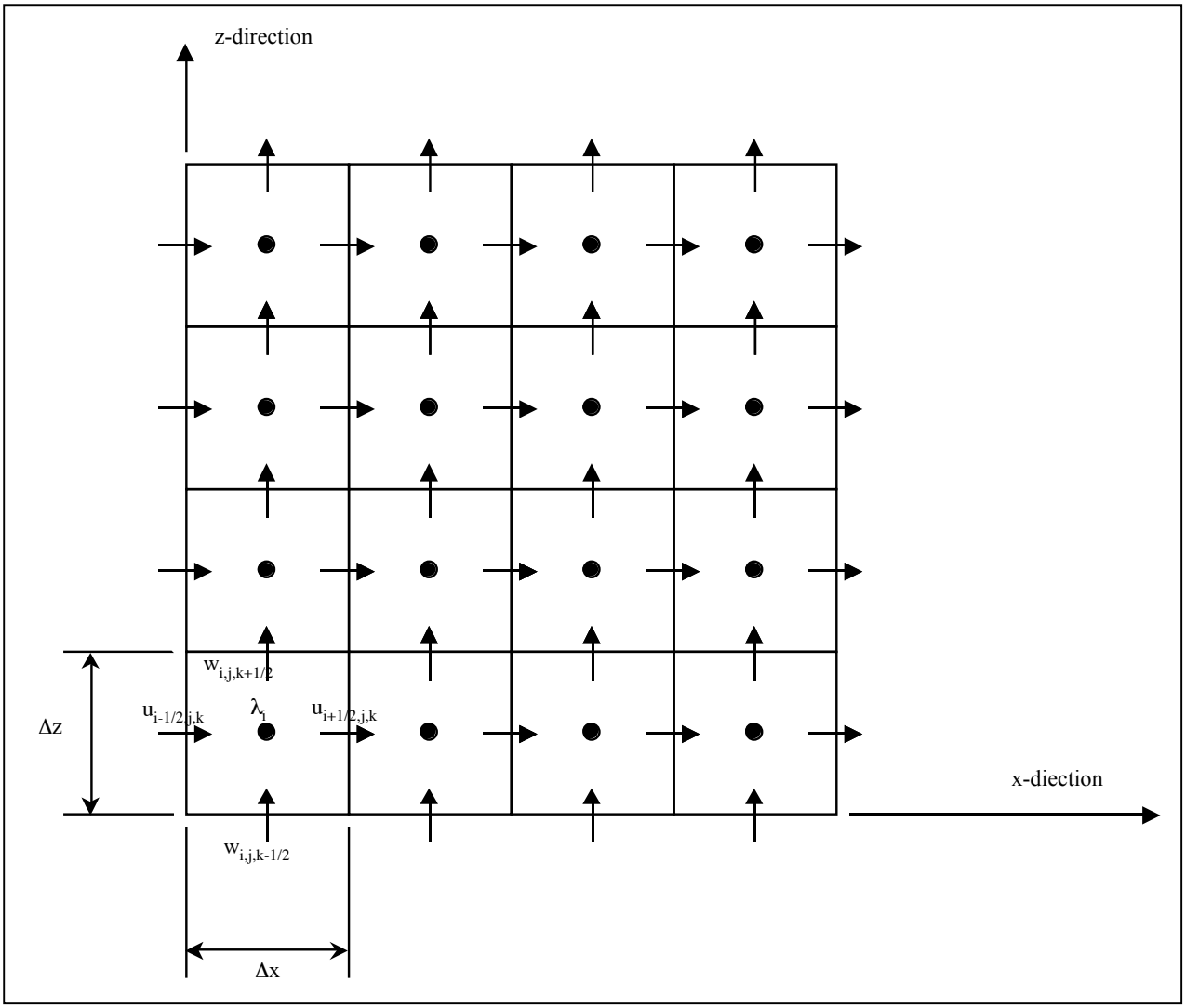


Figure 3 - Staggered grid numerical stencil for QUIC-URB.

Using this type of approximation and substituting $\eta = \left(\frac{\alpha_1}{\alpha_2}\right)^2$ yields,

$$\frac{\lambda_{i-1} - \lambda_i}{\Delta x^2} + \frac{\lambda_{i,j+1,k} - 2\lambda_{i,j,k} + \lambda_{i,j-1,k}}{\Delta y^2} + \eta\left(\frac{\lambda_{i,j,k+1} - 2\lambda_{i,j,k} + \lambda_{i,j,k-1}}{\Delta z^2}\right) = R_{i,j,k} \qquad (6)$$

These type of boundary condition approximation can be easily implemented is several simplifications are made to (2). First multiplying equation (2) by Δx^2

$$\lambda_{i+1,j,k} - 2\lambda_{i,j,k} + \lambda_{i-1,j,k} + \frac{\Delta x^2}{\Delta y^2}\left(\lambda_{i,j+1,k} - 2\lambda_{i,j,k} + \lambda_{i,j-1,k}\right) + \eta\frac{\Delta x^2}{\Delta z^2}\left(\lambda_{i,j,k+1} - 2\lambda_{i,j,k} + \lambda_{i,j,k-1}\right) = \Delta x^2 R_{i,j,k} \qquad (7)$$

Let $A = \frac{\Delta x^2}{\Delta y^2}$ and $B = \eta \frac{\Delta x^2}{\Delta z^2}$. To allow boundary conditions to be implemented the following boundary condition coefficients are used: *e,f,g,h,m,n,o,p,q*. Each coefficient is a cell centered quantity and is varied depending on its location with respect to boundaries.

$$-2o\lambda_{i,j,k} - Ap2\lambda_{i,j,k} - Bq2\lambda_{i,j,k} = \left(\Delta x^2 R_{i,j,k}\right) - \left(e\lambda_{i+1,j,k} + f\lambda_{i-1,j,k} + A\left(g\lambda_{i,j+1,k} + h\lambda_{i,j-1,k}\right) + B\left(m\lambda_{i,j,k+1} + n\lambda_{i,j,k-1}\right)\right) \quad (8)$$

Rearranging and solving for $\lambda_{i,j,k}$

$$\lambda_{i,j,k} = \frac{-\left(\Delta x^2 R_{i,j,k}\right) + \left(e\lambda_{i+1,j,k} + f\lambda_{i-1,j,k} + A\left(g\lambda_{i,j+1,k} + h\lambda_{i,j-1,k}\right) + B\left(m\lambda_{i,j,k+1} + n\lambda_{i,j,k-1}\right)\right)}{2\left(o + Ap + Bq\right)} \quad (9)$$

Equation (5) can easily be put into an SOR form as follows:

$$\lambda^{t+1}{}_{i,j,k} = \omega\left[\frac{-\left(\Delta x^2 R^t{}_{i,j,k}\right) + \left(e\lambda^t{}_{i+1,j,k} + f\lambda^{t+1}{}_{i-1,j,k} + A\left(g\lambda^t{}_{i,j+1,k} + h\lambda^{t+1}{}_{i,j-1,k}\right) + B\left(m\lambda^t{}_{i,j,k+1} + n\lambda^{t+1}{}_{i,j,k-1}\right)\right)}{2\left(o + Ap + Bq\right)}\right] + \left(1-\omega\right)\lambda^{t+1}{}_{i,j,k} \quad (10)$$

where $\omega = 1.78$ is the SOR acceleration parameter. This value was chosen based on the recommendations from Rockle (1990).

The boundary conditions are implemented via the coefficients. Note that each boundary coefficient is unique to each cell so that $e = e_{i,j,k}$.If all of the surrounding cells are fluid cells, then *e,f,g,h,m,n,o,p* and *q* are all unity. If non-fluid cells exist, for example if Equation (4) is to be implemented (a solid wall to the right), the only coefficients that need to be modified from unity are: $o = 0.5$, $e = 0$. Table 1 below enumerates all of the boundary conditions.

Table 1 – A listing of the boundary condition coefficients for QUIC-URB.

No.	E	F	G	H	M	N	O	P	Q	Cell type
1	1	1	1	1	1	0	1	1	0.5	Wall below
2	1	1	1	1	0	1	1	1	0.5	Wall above
3	0	1	1	1	1	1	0.5	1	1	Wall to right
4	1	0	1	1	1	1	0.5	1	1	Wall to left
5	1	1	0	1	1	1	1	0.5	1	Wall behind
6	1	1	1	0	1	1	1	0.5	1	Wall in front
7	1	1	0	1	1	0	1	0.5	0.5	Below & in front
8	1	0	1	1	1	0	0.5	1	0.5	Below & to left
9	0	1	1	1	1	0	0.5	1	0.5	Below & to right
10	1	1	1	0	1	0	1	0.5	0.5	Below & behind

Note that above, below, right, left, front and behind assume that the users is sitting inside a computational cell.

Calculate the divergence of the observed field

The right hand side term in equation (1) is the divergence of the observed flow field:

$$R_{i,j,k} = -2\alpha_1^2\left(\frac{\partial u_o}{\partial x} + \frac{\partial v_o}{\partial y} + \frac{\partial w_o}{\partial z}\right) \tag{11a}$$

This is implemented numerically using the following numerical stencil:

$$R_{i,j,k} = -2\alpha_1^2\left(\frac{u^o{}_{i+\frac{1}{2},j,k} - u^o{}_{i-\frac{1}{2},j,k}}{\Delta x} + \frac{v^o{}_{i,j+\frac{1}{2},k} - v^o{}_{i,j-\frac{1}{2},k}}{\Delta y} + \frac{w^o{}_{i,j,k+\frac{1}{2}} - w^o{}_{i,j,k-\frac{1}{2}}}{\Delta z}\right) \tag{11b}$$

Solve LaPlace's equation

The LaPlace Equation (6) is solved by iterating until a minimum error condition on the Lagrange multiplier is satisfied over a plane in space, namely

$$E_k = \sum_{j=1}^{ny}\sum_{i=1}^{nx}\left|\lambda^{t+1}{}_{i,j,k} - \lambda^t{}_{i,j,k}\right| < \varepsilon \tag{12}$$

Update the new velocity field

The velocity field is updated using the Euler-Lagrange equations whose solution minimizes equation(1).

$$u = u^o + \frac{1}{2\alpha^2{}_1}\frac{\partial\lambda}{\partial x} \tag{13a}$$

$$v = v^o + \frac{1}{2\alpha^2{}_1}\frac{\partial\lambda}{\partial y} \tag{13b}$$

$$w = w^o + \frac{1}{2\alpha^2{}_2}\frac{\partial\lambda}{\partial z} \tag{13c}$$

These equations are subject to the following boundary conditions; $\frac{\partial\lambda}{\partial n} = 0$ at a solid boundary and $\lambda = 0$ at inflow/outflow boundary.

$$\frac{\partial^2\lambda}{\partial x^2} + \frac{\partial^2\lambda}{\partial y^2} + \left(\frac{\alpha_1}{\alpha_2}\right)^2\frac{\partial^2\lambda}{\partial z^2} = -\alpha_1^2\left(\frac{\partial u^o}{\partial x} + \frac{\partial v^o}{\partial y} + \frac{\partial w^o}{\partial z}\right) \tag{14}$$

2.1 Specification of the initial velocity field

The flow field within the entire domain is initialized using a power-law velocity profile:

$$U_o(z) = U_o(z_{ref}) * \left(\frac{z}{z_{ref}} \right)^p \tag{15}$$

The initial flow field around buildings is then specified, as described below.

2.2 Implementation of the Building Flow Parameterizations

i. Flow Regime Determination

The QUIC-URB code first determines the spacing between buildings. If buildings are close together (width-to-height ratio less than two), then algorithms for the initial velocity field are used appropriate to the skimming flow regime. If buildings are spaced farther apart, then algorithms for the an isolated building are used (see Fig. 4).

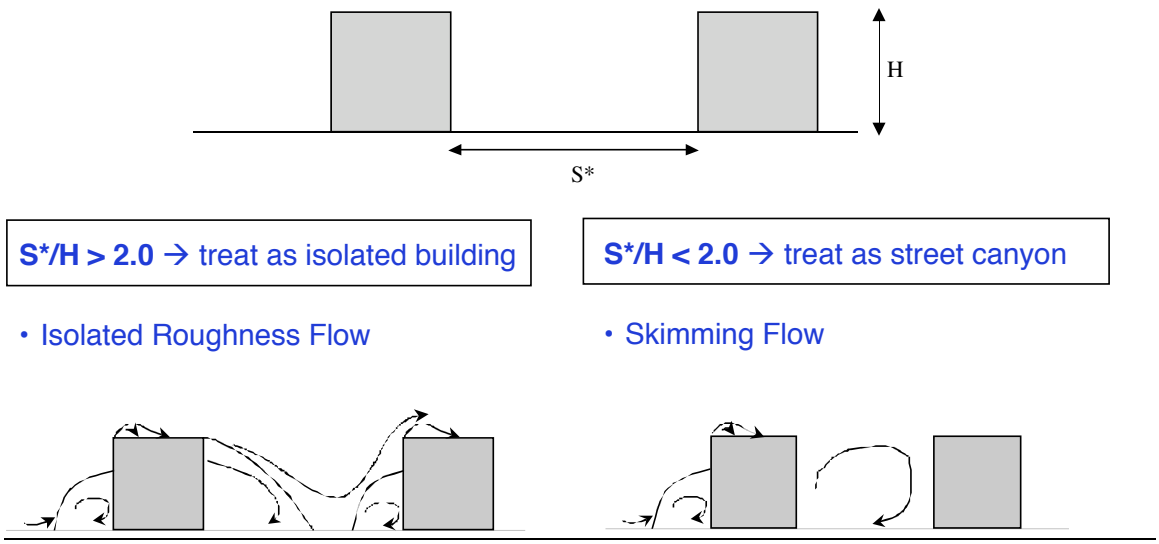


Figure 4. Illustration of the two flow regimes considered in the QUIC-URB model.

ii. Single building parameterizations

a. Primary flow zones

The length of the frontal (L_f) eddy is parameterized using Hosker's (1984) formula:

$$\frac{L_f}{H} = \frac{2\frac{W}{H}}{1 + 0.8\frac{W}{H}}. \tag{16}$$

The length cavity zone (L_r) in the wake of the building is due to Fackrell (1984)

$$\frac{L_r}{H} = \frac{1.8\frac{W}{H}}{\left(\frac{L}{H}\right)^{0.3}\left(1 + 0.24\frac{W}{H}\right)}. \tag{17}$$

Table 2. Velocity specifications in the front cavity:

Front Eddy Zone	FW_x	FW_y
Half axis a_x	$L_f \sin^2\phi\sqrt{1-\left(\frac{z}{0.6h}\right)^2}$	$\frac{l}{2}$
Half axis a_y	$\frac{b}{2}$	$L_f \cos^2\phi\sqrt{1-\left(\frac{z}{0.6h}\right)^2}$
Velocity Specification	$u_o = 0$	$v_o = 0$

Table 3. Dimensions of the windward wake zone and prescription of the velocity components.

Flow Zone	*NN*	*FN*
Half axis a'_x	$L_r\sqrt{1-\left(\frac{z}{h}\right)^2}$	$3L_r\sqrt{1-\left(\frac{z}{h}\right)^2}$
Half axis a'_y	$\frac{b_e}{2}$	$\frac{b_e}{2}$
Velocity Specification	$u_o = -u(h)\left(1-\frac{x_l}{d_N}\right)^2$	$u_o = u(z)\left(1-\frac{x_l}{d_N}\right)^{1.5}$
Velocity Specification	$v_o = -v(h)\left(1-\frac{x_l}{d_N}\right)^2$	$v_o = v(z)\left(1-\frac{x_l}{d_N}\right)^{1.5}$

b. Rooftop and Sidewall recirculation flows

The original Rockle formulation had no rooftop or sidewall recirculation zones.

iii. Multiple building Parameterizations

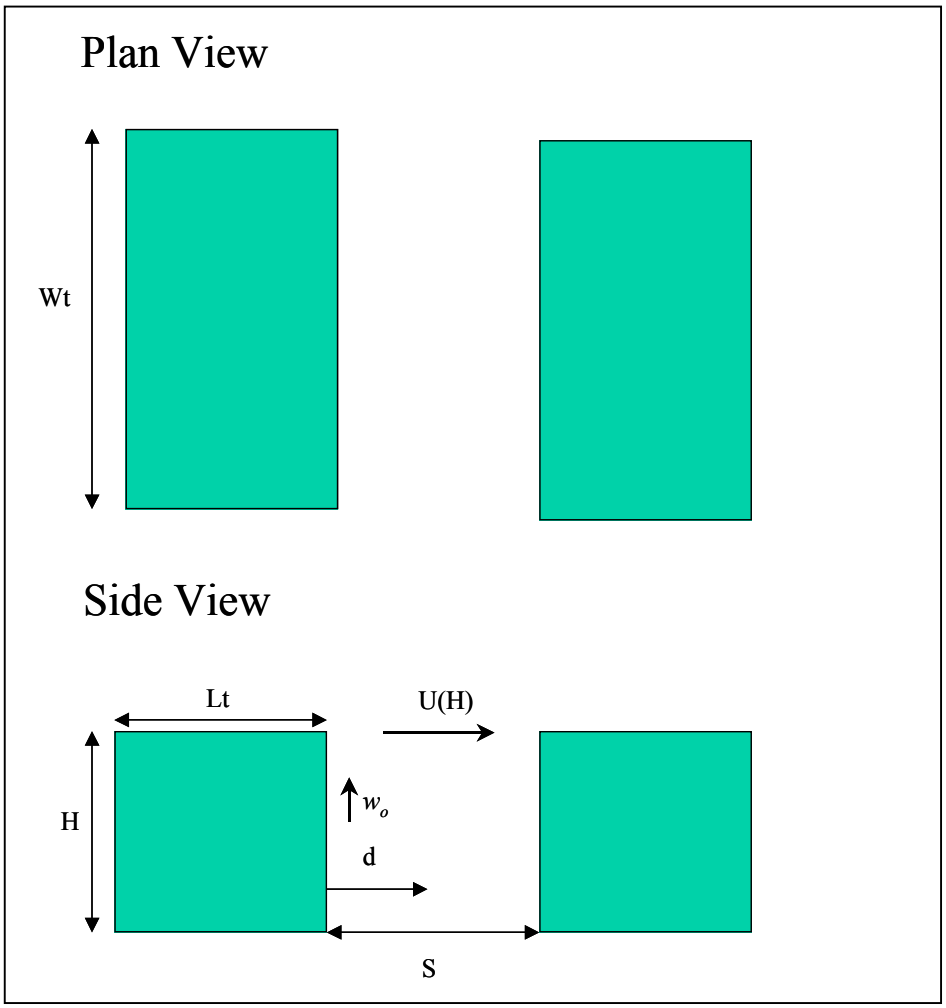
a. Canyon Flows:

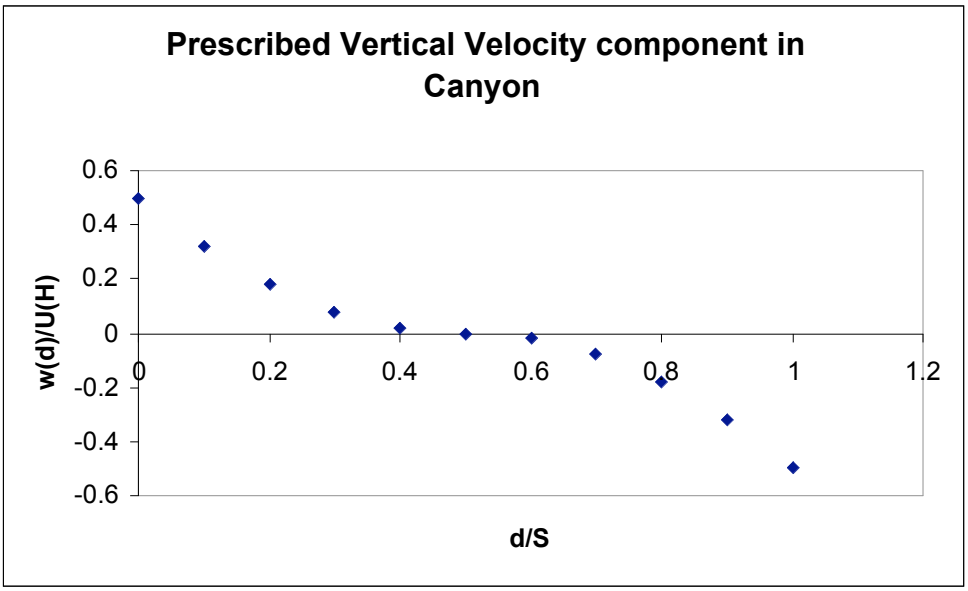
The original Rockle formulation for specification of the initial velocity field between buildings is:

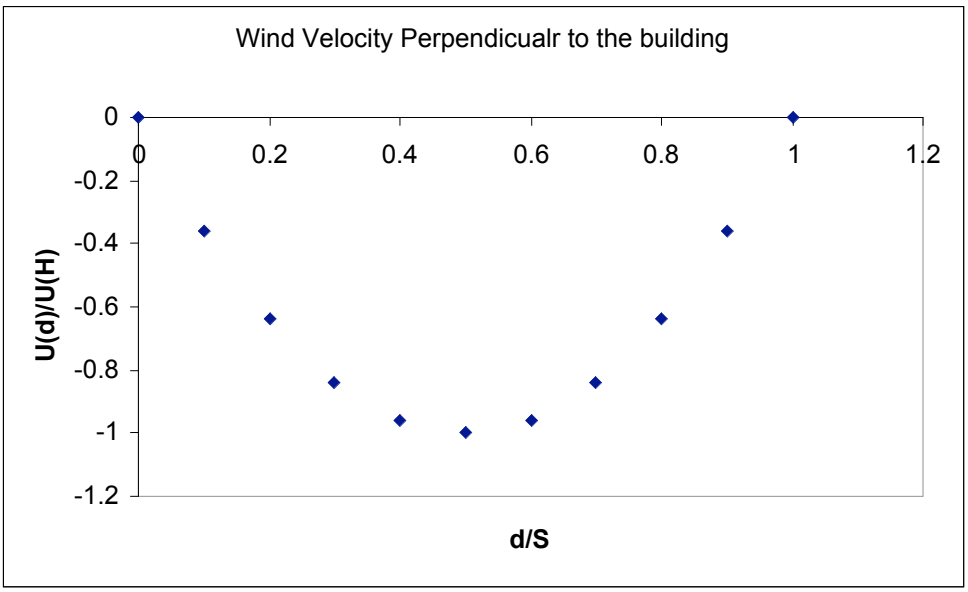
$$\frac{u_0}{U(H)} = -\frac{d}{(0.5S)}\left(\frac{S-d}{0.5S}\right) \tag{18}$$

$$\frac{w_0}{U(H)} = -\left|\frac{1}{2}\left(1-\frac{d}{0.5S}\right)\right|\left(1-\frac{S-d}{0.5S}\right) \tag{19}$$

where S is the spacing between the buildings, d is the distance downwind from the upwind building , w is the vertical velocity component and U(H) is the wind velocity at the top of the upwind building. The figure below shows each of the variables in the canyon. The plots below show the velocity profiles graphically as a function of d/S.




Prescribed Vertical Velocity component in Canyon
w(d)/U(H)
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
0.2
0.4
0.6
0.8
1
1.2
d/S


Wind Velocity Perpendicualr to the building
U(d)/U(H)
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
0
0.2
0.4
0.6
0.8
1
1.2
d/S

b. Flow Above buildings:

The flow above the buildings is implemented using a logarithmic profile in a similar manner as Rockle (1990). In particular (for example See Monin & Obukhov, 1954,)

$$u = \frac{u_*}{k} \ln\left(\frac{z - D}{z_o}\right) \tag{20}$$

where D is displacement height, z_o the aerodynamic roughness and u_* the friction velocity. Since the friction velocity is necessary known the equation is implemented as follows:

$$u = u_{ref} \frac{\ln\left(\frac{z - D}{z_o}\right)}{\ln\left(\frac{z_{ref} - D}{z_o}\right)} \tag{21}$$

where z_{ref} is height of the known measured velocity u_{ref}. The displacement height and roughness length are calculated as follows:

$$D = 0.8 \frac{\sum_{i=1}^{\#blds} W_i L_i h_i}{\sum_{i=1}^{\#blds} W_i L_i} \qquad z_o = 0.2 \frac{\sum_{i=1}^{\#blds} W_i L_i h_i}{L_x L_y} \tag{22}$$

where Lx and Ly are the lengths of the computational domain in the x and y directions.

2.0 Running the Code

In this section the components of the QUIC-URB code are described. In addition, instructions to compile the code using a makefile are described. A description of the input and output files are given along with suggestions for modifying input parameters.

2.1 The Makefile

The QUIC-URB code consists of the following seven subroutines:

```
bcsetup.f        divergence.f     euler.f
init.f           main.f           outfile.f
sor3d.f
```

a makefile and an input file called `input.dat`.

The main program that calls a series of subroutines is `main.f`. The program is compiled using a makefile. Figure 5 is the makefile that has been tested on the following machines: Linux and in a modified form on PGI FORTRAN77 for windows.

```
# makefile for qwicurb.x
# link section ====================================

qwicurb.x: main.o divergence.o sor3d.o euler.o bcsetup.o init.o
outfile.o
      f77 main.o divergence.o sor3d.o euler.o bcsetup.o init.o
outfile.o -o qwicurb.x

#compile section ========================
main.o: main.f
      f77 -c main.f
divergence.o: divergence.f
      f77 -c divergence.f
sor3d.o: sor3d.f
      f77 -c sor3d.f
euler.o: euler.f
      f77 -c euler.f
bcsetup.o: bcsetup.f
      f77 -c bcsetup.f
init.o: init.f
```

Figure 5 – The makefile for QUIC-URB.

The makefile is run by simply typing “make” on the UNIX command line in the directory of the makefile and FORTRAN routines. The name of the executable routine is quicurb.x. With an appropriate input file in place, the code is run by typing “quicurb.x”. The code produces output to the screen as shown below in Figure 6.

```
dx  dy  dz
   1.000000000000000       1.000000000000000       1.000000000000000
 Incident wind angle =    240.0000000000000
 Upstream WS =    1.000000000000000

 building #          1
 Height   Width   Length   xfo   yfo
 10.00    8.00    8.00   15.00   15.00

 upstream winds:
 uo  0.8411271130438497
 vo  0.4856249651385594
 xfo   15.00000000000000       0.0000000000000000E+000inum          1
 Theta =    240.0000000000000       Phi =    30.00000000000000       phip =
  0.5235987755982989
 Weff =    10.92820323027551
 Leff =    10.92820323027551

 cflag_x          0
 cflag_y          0
 fflag          1
 wflag          1
 Lfx =   8.746031743426482
 Lfy =   2.915343914475495
 Lr =   15.17406717554598
 iter        100
 iter        200
 iter        300
 iter        400
 iteration =         402
```

Figure 7 - Screen output displayed while running QUIC-URB.

The screen output shown in Figure 7 is for a single building case. Here Leff and Weff are the effective length and width seen by the wind, Lfx and Lfy are the sizes of the upwind cavities, Lr is the length of recirculation cavity, cflag_x and cflag_y are street canyon flags indicating weather a street canyon vortex is being placed behind the building. Fflag indicates that a front eddy is being placed upwind of the building and wflag indicates that the wake is being added to the rear of the building. The last line indicates the number of iterations necessary for convergence in SOR solver.

2.2 The Input File

Running QUIC-URB requires one input file. The text of the input file "input.dat" is shown below in Figure 8.

```
270.                    !wind angle (std met, ie 0 out of North, 90 out of East
1.                      !wind speed  (meters/sec)
6.                      !reference height for power law
0.16                    !exponent for inlet profile
1                       !building input mode (1 for input, 2 for city file)
2                       !number of buildings
bldnum  bldtype   height width   length  xfo     yfo
1       1         19      6       6       15      12
2       1         19      6       6       30      12
```

Figure 8 - Input data file – input.dat – for QUIC-URB.

The first line of the input file is the approach wind angle in degrees, in standard meteorological format. For example, wind from the east would be 90 and wind from the west 270. The second line is simply the wind speed at reference height in meters/second. The third line of the code is the reference height for the upwind boundary layer velocity profile and the fourth line is the exponent. That is:

$$\frac{u_o(z)}{u_{ref}} = \left(\frac{z}{z_{ref}} \right)^p$$

The fifth line of the file indicates weather building data will be entered in the input file or through a building data base file. Currently only option 1 is available. Line six defines the number of buildings that will be used in the calculations. Line seven is a text line that is not used in the code. Lines 8 and greater specify the building geometry and locations. The width is the dimension of the building in the y-direction, the length the dimension of the building in the x-direction and height is length of the building in the z-direction. The "front-center" of the building is defined by looking at a plan view as shown below in the figure.

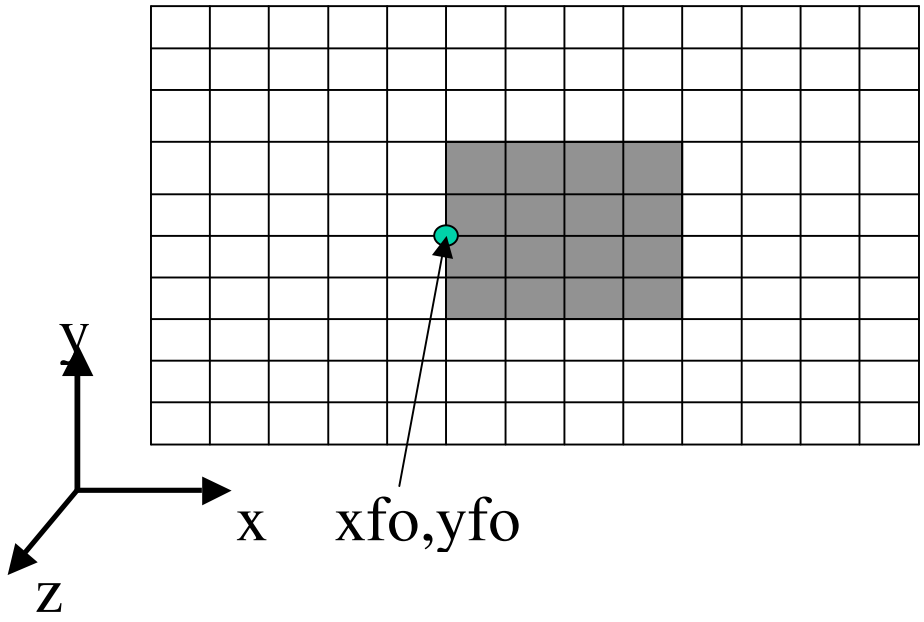


Figure 9 – Plan view of a building in the QUIC-URB coordinate system showing the building front/center points xfo and yfo.

2.3 Output Data Files

The users can modify the source code to produce any output they want, however the standard version of QUIC-URB has the following output files:

`celltype.dat`	`celltype2.dat`	`uofield.dat`
`uoutmat.dat`	`uoutfield.dat`	`buildout.dat`

All output files are currently in ASCII format. The `celltype.dat` file and `celltype2.dat` files contains a gridded data set for the "ground" and "building" respectively. This file is useful for visualizing the location of the buildings. This is the used in the Matlab post processing GUI for generating buildings. The files contain x,y,z and a building flag. The building flag is set to 0 for a fluid cell and 1 for a solid cell. The `uofield.dat` has the initial velocity data before the mass conservation step of the code. That is, it contains the data from the raw empirical parameterizations. The files `uoutmat.dat` and `uoutfield.dat` are essentially the same (that is they contain: x,y,z,u,v,w data in ASCII format), however `uoutmat.dat` has a header format easily read into the Matlab GUI, while `uoutfield.dat` has a header for being read into Tecplot. An example of a streamline plot is shown below in Figure 10.

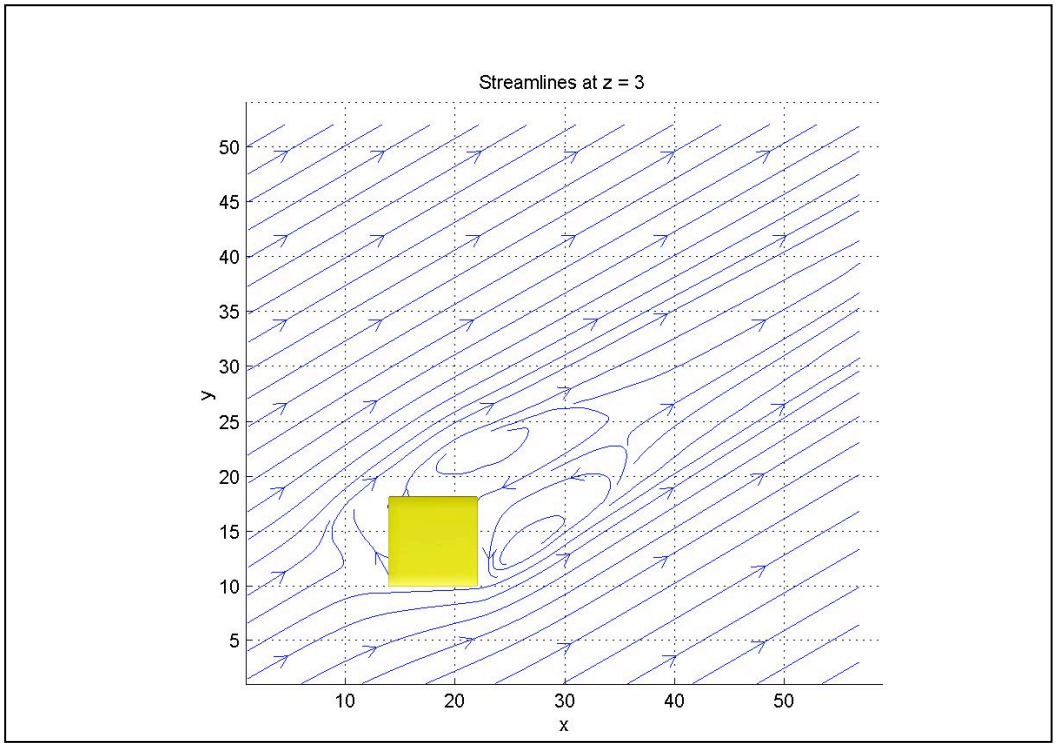


Figure 10 – Streamlines of flow out the south west around a single cube using the Matlab QUIC-URB GUI visualization tool.

2.4 Choosing the Input Parameters

There are various parameters that need to be entered into the `input.dat` file. Wind angle input is in standard meteorological where winds are specified with reference to true north. For example a wind out of the south would be input as 180 degrees. Wind speeds should be entered in meters per second and all building dimensions should be in meters.

3.0 Description of the Program and Subroutines

QUIC-URB contains a main program and six subroutines. Within the source code itself, there is a brief description of what each subroutine does along with comments within the code. Below is a brief description of the main program and each of the subroutines.

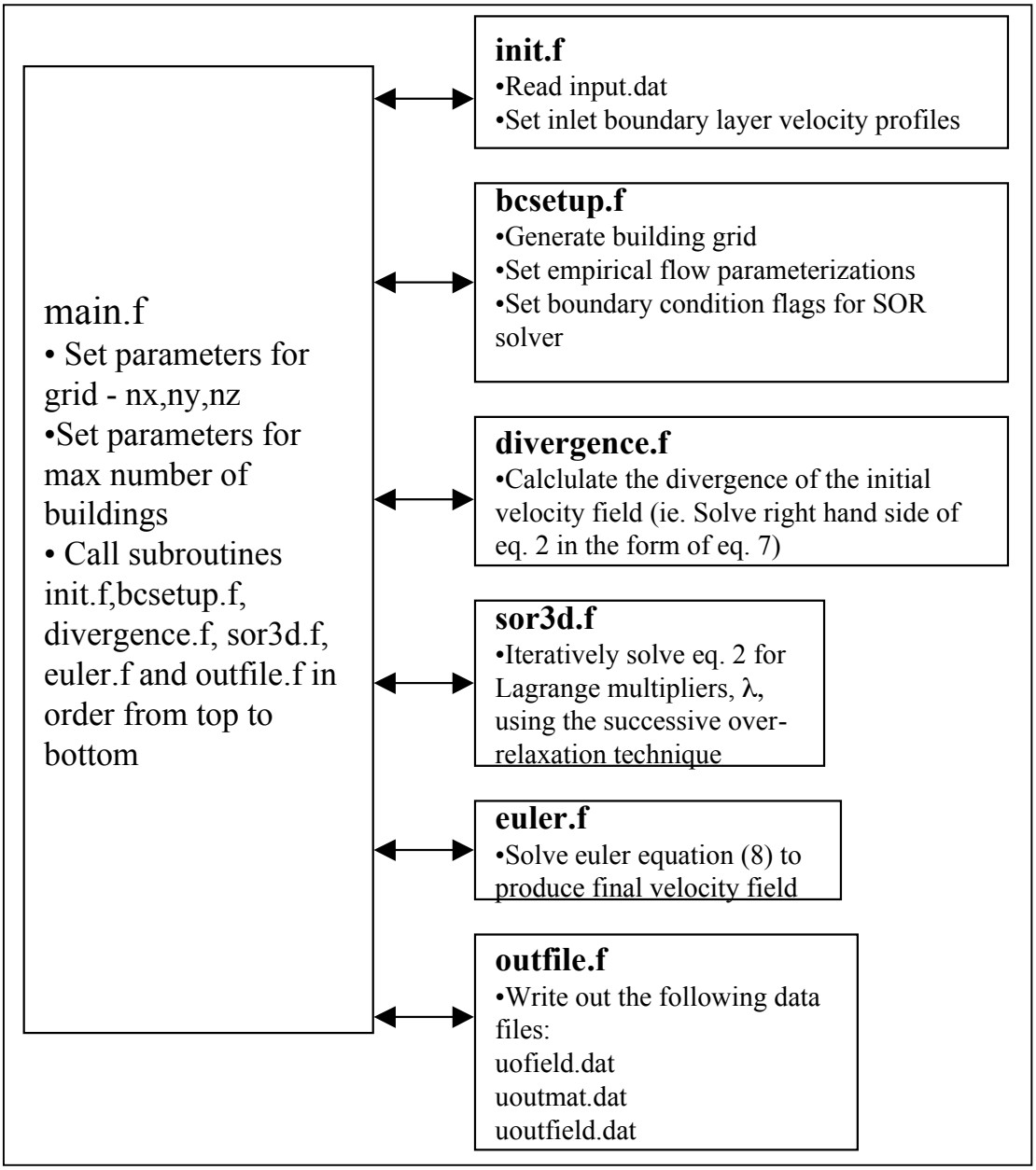
3.1 Program main.f

Program `main.f` is fairly short, but performs several tasks. It calls all of the subroutines (`init.f`, bcsetup.f, `divergence.f`, `sor3d.f`, `euler.f`, and `outfile.f`). Several important parameter statements are set in main.f, they include: `nx`, `ny` and `nz`, the number of grid vertices in the x, y and z directions. There is also a parameter statement

for the maximum number of buildings called `maxnumbuild`. The maximum number of iterations allowed in the SOR solver is also set here (variable *itermax*) as well as the convergence criteria for the solver (i.e., the residual variable *eps*). Variables are read from `input.dat` in `init.f`. The initial wind direction vectors (u_o, v_o and w_o) are also set here. The output files `uofield.dat`, `uoutmat.dat` and `uoutfield.dat` are written out in `outfile.f`. The bulk of the computational work is done in bcsetup.f, sor3d.f and euler.f. Refer to the theory section of the manual for the equation that are solved. Figure 6 is a flowchart showing the each of the subroutines with a small description. Note that main.f calls each of the subroutines in order from top to bottom as shown in Figure 6.

3.2 Subroutine init.f

Variables are declared, read from `input.dat` and set in `init.f`. The initial wind direction vectors and inlet boundary layer velocity profile are also set here.



3.3 Subroutine bcsetup.f

This subroutine handles the bulk of the initializations necessary to perform the mass consistent calculations. The subroutine begins by generating the locations of solid cells that indicate the location of the ground and buildings. The next step involves implementing the various "urban parameterizations" for the velocity field depending on the number of buildings, wind angle and building spacing.

This subroutine also handles setting the boundary condition flags for the SOR solver as discussed above in section 2.1.

3.4 Subroutine divergence.f

This subroutine calculates the divergence of the initial velocity field as per equation (7) in Section 2.

3.5 Subroutine sor3d.f

This subroutine is the Successive Over-Relaxation (SOR) solving subroutine. In this subroutine, the Poisson's equation for the Lagrange multipliers is solved for using a central difference stencil (See equation 6) by iterating until the point-wise error is reduced below 1e-6 or until 10,000 iterations are made.

3.6 Subroutine euler.f

This subroutine updates the final velocity field *u,v* and *w* based on equation 8.

3.7 Subroutine outfile.f

This subroutine simply writes of the following data files in ASCII format: uoutmat.dat uoutfield.dat and uofield.dat. Note that the headers written to files uofield.dat and uoutfield.dat are specific to Tecplot users, while uoutmat is specific to Matlab users.

4.0 Future Work – Model improvements

Any comments or corrections to the code or this document would be appreciated. Please send any comments or corrections to: pardyjak@eng.utah.gov

5.0 Acknowledgments

This project was funded by the Department of Energy as part of the Chemical Biological Non-Proliferation (CBNP) program.

6.0 References

Kaplan, H. and N. Dinar. A Lagrangian Dispersion Model for Calculating Concentration Distribution within a built-up domain, *Atmos. Env.*, Vol. 30, No. 24, pp. 4197-4207, 1996.

Dinar, N., Mass Consistent Models for Wind Distribution in Complex Terrain – Fast Algorithms for Three Dimensional Problems, *Boundary-Layer Met.* Vol. 30, pp. 177-199 1984.

Röckle, R. (1990) Bestimmung der Stomungsverhaltnisse im Bereich komplexer Bebauungsstrukturen. Ph.D. thesis, Vom Fachbereich Mechanik, der Technischen Hochschule Darmstadt, Germany.

Sherman, C. A., A Mass Consistent Model for Wind Fields over Complex Terrain, Journal of Applied Meteorology, vol. 17, 1978.